Attorney Docket No. 223563
MS # 302409.01

PATENT APPLICATION

Invention Title:

METHOD OF NEGOTIATING SECURITY PARAMETERS AND AUTHENTICATING USERS INTERCONNECTED TO A NETWORK

Inventors:

Brian D. Swander	US	Bellevue	Washington
INVENTOR'S NAME	CITIZENSHIP	CITY OF RESIDENCE	STATE or FOREIGN COUNTRY

Sara Bitan	Israel	Moshav Hadar-AM	Israel
INVENTOR'S NAME	CITIZENSHIP	CITY OF RESIDENCE	STATE or FOREIGN COUNTRY

Christian Huitema	France	Clyde Hill	WAshington
INVENTOR'S NAME	CITIZENSHIP	CITY OF RESIDENCE	STATE or FOREIGN COUNTRY

Paul G. Mayfield	US	Sammamish	Washington
INVENTOR'S NAME	CITIZENSHIP	CITY OF RESIDENCE	STATE or FOREIGN COUNTRY

Daniel R. Simon	Canada	Redmond	Washington
INVENTOR'S NAME	CITIZENSHIP	CITY OF RESIDENCE	STATE or FOREIGN COUNTRY

Be it known that the inventors listed above have invented a certain new and useful invention with the title shown above of which the following is a specification.

METHOD OF NEGOTIATING SECURITY PARAMETERS AND AUTHENTICATING USERS INTERCONNECTED TO A NETWORK

FIELD OF THE INVENTION

This invention generally relates to the area of computer systems. More particularly, the present invention concerns methods for facilitating the use of a security protocol to protect network communications, and even more particularly to methods for negotiating security parameters and authenticating users interconnected to a network.

BACKGROUND OF THE INVENTION

Computer networks provide an efficient way to exchange information between two or more computers. Various types of computer networks are utilized including private networks, e.g. a local area networks (LANs), and public networks, e.g. the Internet. Often, the information exchanged between computers is of a sensitive or confidential nature. For example, to purchase goods or services via the network, a user is required to enter payment information such as a credit card number. Similarly, users routinely transmit sensitive and confidential business information over networks.

Information is exchanged over networks according to a protocol, such as the Internet Protocol (IP). IP was designed to allow for an open exchange of information; however, standard IP was not designed to protect information from unauthorized access. Accordingly, standard IP does not prevent an unauthorized user from receiving, viewing, and even modifying information transmitted over a network. Standard IP lacks other features such as authentication of users and network devices.

To address the lack of security provided by standard IP, the Internet Engineering Task Force (IETF) has developed a set of protocols, referred to as the Internet Protocol Security (IPSec) suite. IPSec provides protocols that conform to standard IP, but that include security features lacking in standard IP. Specific examples of IPSec protocols include an authentication header (AH) protocol and encapsulating security protocol (ESP). The ESP protocol, documented mainly in IETF Request for Comments (RFC) 2406, is an authenticating and encrypting protocol that uses cryptographic mechanisms to provide integrity, source authentication, and confidentiality of data. The AH protocol, documented mainly in IETF RFC 2402, is an authentication protocol that uses a hash

signature in the packet header to validate the integrity of the packet data and authenticity of the sender.

Prior to using the ESP, AH or similar protocols, a first computer and a second computer in communication over the network must negotiate a set security parameters. The first computer begins the negotiation and is usually referred to as an initiator. The second computer is referred to as a responder because it is responding to a request from the initiator. The negotiated security parameters are stored in the initiator and the responder as one or more data structures referred to as a security association (SA). Parameters stored in the SA identify a security protocol (e.g. ESP or AH), a cryptographic algorithm used to secure communication (e.g. DES, 3DES), keys used with the cryptographic algorithm, a life time during which the keys are valid and the like.

One method of negotiating security parameters is by using a separate negotiation protocol. An example of a negotiation protocol is the internet key management and exchange protocol (IKE), also provided as part of IPSec and documented in IETF RFC 2409. The IKE protocol includes two phases. In a first phase, known as "main mode," the initiator and the responder establish an IKE SA thereby creating a secure channel for conducting IKE negotiations. In a second phase, known as "quick mode," the initiator and the responder use the IKE SA to negotiate general purpose SAs over the secure channel established in the first phase.

An IKE negotiation can fail for various reasons. As one example, the initiator and the responder can fail to agree on an acceptable set of security parameters. The initiator can attempt a new IKE negotiation by proposing different security parameters. However, IKE does not provide a mechanism for the initiator to predict whether the responder will accept the different set of proposed parameters. Accordingly, the new IKE negotiation may likewise fail.

Moreover, IKE provides for machine authentication, but not user authentication. Thus, while it is possible to verify the identity of a particular machine, it is not possible to verify the identity of a particular user. Some methods have been developed to incorporate user authentication into IKE using other known protocols such as Kerberos. However, these methods require that a new IKE main mode be conduced in conjunction with each user authentication.

Compatibility issues also exist when some protocols are combined with IKE. For example, when the initiator sends a request to the responder in clear text, meaning not according to a security protocol, and the responder requires secure communication, the responder initiates an IKE negotiation. When this occurs, the responder effectively becomes the initiator and the initiator effectively becomes the responder thereby subverting the roles of the initiator and the responder. Protocols, such as Kerberos, are sensitive to the direction of the negotiation and can fail when the roles of initiator and responder are subverted.

SUMMARY OF THE INVENTION

The present invention comprises a method for negotiating security parameters between a first computer, called an initiator, and a second computer, called a responder, interconnected to a network. The method includes both user and machine authentication. The method has a plurality of modes including a main mode and a quick mode conducted through the exchange of a plurality of messages between the initiator and the responder.

The main mode is used to perform machine one way or mutual machine authentication and to provide a secure channel for conducting the quick mode and user mode. One way machine authentication is used to prove the identity of one of, but not both, the initiator and the responder. Mutual machine authentication is used to prove the identity of both the initiator and the responder. The quick mode is used to derive and refresh keys used with IPSec protocols such as ESP and AH.

The invention further comprises an optional user mode. The user mode provides one way or mutual user authentication. One way user authentication is used to prove the identity of a particular user of one of, but not both, the initiator and the responder. Mutual user authentication is used to prove the identity of the users of both the initiator and responder. A plurality of user modes can be carried out following a single main mode.

The invention also provides for policy discoverability allowing the initiator to learn acceptable security policy of the responder and vice versa. Group data may optionally be exchanged between the initiator and responder, which data can be compared to a set of authorized groups to determine if communication is permitted and, if

so, to select policy used during the communication. Additional features and advantages of the invention will be made apparent from the following detailed description of illustrative embodiments, which proceeds with reference to the accompanying figures.

BRIEF DESCRIPTION OF THE DRAWINGS

While the appended claims set forth the features of the present invention with particularity, the invention, together with its objects and advantages, is best understood from the following detailed description taken in conjunction with the accompanying drawings of which:

FIG. 1 is a simplified schematic illustrating an exemplary architecture of a network device for carrying out a method in accordance with an embodiment of the present invention;

FIG. 2 is an exemplary network environment including multiple network devices coupled to a network;

FIG. 3 is a simplified diagram of a packet payload format used to exchange payload data;

FIG. 4 is a diagram illustrating a method of conducting main mode and quick mode negotiations;

FIG. 5 is a diagram illustrating a method of initiating a main mode that provides legacy coexistence with prior negotiating protocols;

FIG. 6 is a diagram illustrating a method of conducting a user mode;

FIG. 7 is a diagram illustrating a method of dynamically discovering policy of a network device;

FIG. 8 is a diagram illustrating a method of dynamically discovering policy of a network device during a secure negotiation;

FIG. 9 is a diagram illustrating a method of negotiating security parameters using a group identification;

FIG. 10 is a diagram illustrating a method of negotiating security parameters using a previously identified public key; and

FIG. 11 is a flow diagram illustrating a method used by an initiator to conduct a security negotiation.

DETAILED DESCRIPTION OF THE DRAWINGS

Turning to the drawings, wherein like reference numerals refer to like elements, the invention is illustrated as being implemented in a suitable computing environment. Although not required, the invention will be described in the general context of computer-executable instructions, such as program modules, being executed by a network device, such as a personal computer. Generally, program modules include routines, programs, objects, components, data structures, etc. that perform particular tasks or implement particular abstract data types. Moreover, those skilled in the art will appreciate that the invention may be practiced with other computer system configurations, including hand-held devices, multi-processor systems, microprocessor based or programmable consumer electronics, network PCs, minicomputers, mainframe computers, and the like. The invention may also be practiced in distributed computing environments where tasks are performed by remote processing devices that are linked through a communications network. In a distributed computing environment, program modules may be located in both local and remote memory storage devices.

FIG. 1 illustrates an example of a suitable computing system environment 100 on which the invention may be implemented. The computing system environment 100 is only one example of a suitable computing environment and is not intended to suggest any limitation as to the scope of use or functionality of the invention. Neither should the computing environment 100 be interpreted as having any dependency or requirement relating to any one or combination of components illustrated in the exemplary operating environment 100.

The invention is operational with numerous other general purpose or special purpose computing system environments or configurations. Examples of well known computing systems, environments, and/or configurations that may be suitable for use with the invention include, but are not limited to: personal computers, server computers, hand-held or laptop devices, tablet devices, multiprocessor systems, microprocessor-based systems, set top boxes, programmable consumer electronics, network PCs, minicomputers, mainframe computers, distributed computing environments that include any of the above systems or devices, and the like.

The invention may be described in the general context of computer-executable instructions, such as program modules, being executed by a computer. Generally, program modules include routines, programs, objects, components, data structures, etc. that perform particular tasks or implement particular abstract data types. The invention may also be practiced in distributed computing environments where tasks are performed by remote processing devices that are linked through a communications network. In a distributed computing environment, program modules may be located in local and/or remote computer storage media including memory storage devices.

With reference to **FIG. 1**, an exemplary system for implementing the invention includes a general purpose computing device in the form of a computer 110. Components of the computer 110 may include, but are not limited to, a processing unit 120, a system memory 130, and a system bus 121 that couples various system components including the system memory to the processing unit 120. The system bus 121 may be any of several types of bus structures including a memory bus or memory controller, a peripheral bus, and a local bus using any of a variety of bus architectures. By way of example, and not limitation, such architectures include Industry Standard Architecture (ISA) bus, Micro Channel Architecture (MCA) bus, Enhanced ISA (EISA) bus, Video Electronics Standards Association (VESA) local bus, and Peripheral Component Interconnect (PCI) bus also known as Mezzanine bus.

The computer 110 typically includes a variety of computer readable media. Computer readable media can be any available media that can be accessed by the computer 110 and includes both volatile and nonvolatile media, and removable and non-removable media. By way of example, and not limitation, computer readable media may comprise computer storage media and communication media. Computer storage media includes volatile and nonvolatile, removable and non-removable media implemented in any method or technology for storage of information such as computer readable instructions, data structures, program modules or other data. Computer storage media includes, but is not limited to, RAM, ROM, EEPROM, flash memory or other memory technology, CD-ROM, digital versatile disks (DVD) or other optical disk storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, or any other medium which can be used to store the desired information and

which can be accessed by the computer 110. Communication media typically embodies computer readable instructions, data structures, program modules or other data in a modulated data signal such as a carrier wave or other transport mechanism and includes any information delivery media. The term "modulated data signal" means a signal that has one or more of its characteristics set or changed in such a manner as to encode information in the signal. By way of example, and not limitation, communication media includes wired media such as a wired network or direct-wired connection, and wireless media such as acoustic, RF, infrared and other wireless media. Combinations of the any of the above should also be included within the scope of computer readable media.

The system memory 130 includes computer storage media in the form of volatile and/or nonvolatile memory such as read only memory (ROM) 131 and random access memory (RAM) 132. A basic input/output system 133 (BIOS), containing the basic routines that help to transfer information between elements within computer 110, such as during start-up, is typically stored in ROM 131. RAM 132 typically contains data and/or program modules that are immediately accessible to and/or presently being operated on by processing unit 120. By way of example, and not limitation, Figure 1 illustrates operating system 134, application programs 135, other program modules 136 and program data 137.

The computer 110 may also include other removable/non-removable, volatile/nonvolatile computer storage media. By way of example only, **FIG. 1** illustrates a hard disk drive 141 that reads from or writes to non-removable, nonvolatile magnetic media, a magnetic disk drive 151 that reads from or writes to a removable, nonvolatile magnetic disk 152, and an optical disk drive 155 that reads from or writes to a removable, nonvolatile optical disk 156 such as a CD ROM or other optical media. Other removable/non-removable, volatile/nonvolatile computer storage media that can be used in the exemplary operating environment include, but are not limited to, magnetic tape cassettes, flash memory cards, digital versatile disks, digital video tape, solid state RAM, solid state ROM, and the like. The hard disk drive 141 is typically connected to the system bus 121 through a non-removable memory interface such as interface 140, and magnetic disk drive 151 and optical disk drive 155 are typically connected to the system bus 121 by a removable memory interface, such as interface 150.

The drives and their associated computer storage media, discussed above and illustrated in **FIG. 1**, provide storage of computer readable instructions, data structures, program modules and other data for the computer 110. In **FIG. 1**, for example, hard disk drive 141 is illustrated as storing operating system 144, application programs 145, other program modules 146 and program data 147. Note that these components can either be the same as or different from operating system 134, application programs 135, other program modules 136, and program data 137. Operating system 144, application programs 145, other program modules 146, and program data 147 are given different numbers hereto illustrate that, at a minimum, they are different copies. A user may enter commands and information into the computer 110 through input devices such as a tablet, or electronic digitizer, 164, a microphone 163, a keyboard 162 and pointing device 161, commonly referred to as a mouse, trackball or touch pad. Other input devices (not shown) may include a joystick, game pad, satellite dish, scanner, or the like. These and other input devices are often connected to the processing unit 120 through a user input interface 160 that is coupled to the system bus, but may be connected by other interface and bus structures, such as a parallel port, game port or a universal serial bus (USB). A monitor 191 or other type of display device is also connected to the system bus 121 via an interface, such as a video interface 190. The monitor 191 may also be integrated with a touch-screen panel or the like. Note that the monitor and/or touch screen panel can be physically coupled to a housing in which the computing device 110 is incorporated, such as in a tablet-type personal computer. In addition, computers such as the computing device 110 may also include other peripheral output devices such as speakers 197 and printer 196, which may be connected through an output peripheral interface 194 or the like.

The computer, or network device, 110 operates in a networked environment using logical connections to one or more remote computers, such as a remote computer 180. The remote computer 180 may be a personal computer, a server, a router, a network PC, a peer device or other common network node, and typically includes many or all of the elements described above relative to the computer 110, although only a memory storage device 181 has been illustrated in **FIG. 1**. The logical connections depicted in **FIG. 1** include a local area network (LAN) 171 and a wide area network (WAN) 173, but may

also include other networks. Such networking environments are commonplace in offices, enterprise-wide computer networks, intranets and the Internet. For example, in the present invention, the computer system 110 may comprise the source machine from which data is being migrated, and the remote computer 180 may comprise the destination machine. Note however that source and destination machines need not be connected by a network or any other means, but instead, data may be migrated via any media capable of being written by the source platform and read by the destination platform or platforms.

When used in a LAN networking environment, the computer 110 is connected to the LAN 171 through a network interface or adapter 170. When used in a WAN networking environment, the computer 110 typically includes a modem 172 or other means for establishing communications over the WAN 173, such as the Internet. The modem 172, which may be internal or external, may be connected to the system bus 121 via the user input interface 160 or other appropriate mechanism. In a networked environment, program modules depicted relative to the computer 110, or portions thereof, may be stored in the remote memory storage device. By way of example, and not limitation, **FIG. 1** illustrates remote application programs 185 as residing on memory device 181. It will be appreciated that the network connections shown are exemplary and other means of establishing a communications link between the computers may be used.

In the description that follows, the invention will be described with reference to acts and symbolic representations of operations that are performed by one or more computers, unless indicated otherwise. As such, it will be understood that such acts and operations, which are at times referred to as being computer-executed, include the manipulation by the processing unit of the computer of electrical signals representing data in a structured form. This manipulation transforms the data or maintains it at locations in the memory system of the computer, which reconfigures or otherwise alters the operation of the computer in a manner well understood by those skilled in the art. The data structures where data is maintained are physical locations of the memory that have particular properties defined by the format of the data. However, while the invention is being described in the foregoing context, it is not meant to be limiting as those of skill in the art will appreciate that various of the acts and operation described hereinafter may also be implemented in hardware.

FIG. 2 illustrates an exemplary network environment wherein the present invention is employed. The present invention is directed to a method for negotiating security parameters between networks devices in communication through one or more networks. The invention also provides features such as policy discoverability and mutual or one way machine and user authentication. The invention is implemented as an extension to existing protocols, such as the Internet key exchange and management protocol (IKE). Alternatively, the invention is implemented as a separate proprietary protocol.

The environment includes a plurality of network devices 202, 204, 206 communicatively coupled to a network 208. The network 208 is any suitable type such as a local area network (LAN), wide area networks (WAN), intranet, the Internet, or any combination thereof. For the purpose of illustrating the invention, only a limited number of network devices are shown. However, it will be understood that many network devices may, in fact, be coupled to the network. Moreover, although the network devices are illustrated as coupled directly to the network 208, the network devices are alternatively coupled to the network 208 through a combination of servers, routers, proxies, gateways, network address translation devices, or the like.

The network device 202 communicates, i.e. exchanges information, with the network device 204 by sending packets of data according to a protocol such as the Internet Protocol (IP). The network device 202, referred to herein as the initiator, begins the exchange of information by sending a request to the network device 204, referred to herein as the responder. The network device 206 is a malicious user that attempts to gain unauthorized access to the information exchanged between the initiator 202 and the responder 204. The malicious user 206 also attempts to mount attacks on one or more of the initiator 202 and the responder 204 through, for example, a denial of service attack.

The initiator 202 includes a security policy 216 stored in security policy data base. The security policy 216 is used by the initiator 202 to determine whether data transmitted to, or received from, another network device, such as the responder 204, needs to conform to a security protocol such as the Encapsulating Security Protocol (ESP) or Authentication Header (AH). The responder 204 includes its own security policy 220 stored in a security policy database that is used by the responder 204 to determine

whether data transmitted to, or received from, another device, such as the initiator 202, needs to conform to a security protocol.

Security protocols such as AH and ESP protect the contents of data in an IP packet from the attacker 206. Before the security protocol is used to exchange data, the initiator 202 and the responder 204 must negotiate security parameters. The negotiated security parameters include an identification of the security protocol to be used (e.g. AH or ESP), an encryption algorithm that will be used to secure the data (e.g. DES or 3DES), keys used with the encryption algorithm to protect the data, life time that the keys will be valid and the like. The negotiated security parameters are stored in one or more data structures called a Security Association (SA).

The invention provides a method to negotiate the security parameters. The method includes a plurality of modes including a main mode 210, a quick mode 212, and a user mode 214. A negotiation between the initiator 202 and the responder 204 requires at least one main mode 210 and at least one quick mode 212. The user mode 214 is not required and is optionally conducted when user authentication is desired. Further, a one to one correspondence between the plurality of modes is not required. For example, a single main mode 210 supports a plurality of user modes 212 and a plurality of quick modes 212. A single user mode 214 supports a plurality of quick modes 212.

The method is executed by negotiation module 218 executing in the initiator 202 and negotiation module 222 executing in the responder 204. Each of the plurality of modes includes one or more pair of exchanges between the initiator 202 and the responder 204. Each exchange includes a first message sent from the initiator 202 to the responder 204 and a second message sent from the responder 204 to the initiator 202.

The main mode 210 is used to perform machine one way or mutual machine authentication, negotiate security parameters, and to provide a secure channel for conducting the quick mode 212 and user mode 214. One way machine authentication is used to prove the identity of one of, but not both, the initiator 202 and the responder 204. Mutual machine authentication is used to prove the identity of both the initiator 202 and the responder 204.

The quick mode 212 is used to derive and refresh keys used with IPSec protocols such as ESP and AH. Typically, the keys used with AH and ESP to encrypt and decrypt

data have a limited life time defined by the SA, referred to as life time of the key. Thus, it necessary for the initiator 202 and responder 204 to periodically refresh keys used as part of the security protocol. Keys are refreshed by executing a new quick mode 212.

The user mode 214 provides one way or mutual user authentication. One way user authentication is used to prove the identity of a particular user of one of, but not both, the initiator 202 and the responder 204. Mutual user authentication is used to prove the identity of the users of both the initiator 202 and responder 204. Multiple users may be associated with a particular network device. For example, a plurality of users are associated with the initiator 202. Thus, according to the invention, a single main mode 210 is used to authenticate the network device of the initiator 202 to the responder 204 thereby providing machine authentication. The plurality of users are authenticated to the responder 204 by executing a plurality of user modes 214.

FIG. 3 illustrates an example of a packet 228, referred to herein as a message, used to exchange data between the initiator 202 and the responder 204. The packet or message 228 includes a header portion 230 and one or more payloads 232. The format illustrated in **FIG. 3** generally conforms to the IKE protocol. It will be understood that the format described is by way of example, and not limitation, as any suitable format can be used to exchange data between the initiator 202 and the responder 204.

The header 230 includes an Initiator Cookie (I-Cookie) 236 and a Responder Cookie (R-Cookie) 238. The I-Cookie is a non-zero value assigned by the initiator 202 and the R-Cookie is a non-zero value assigned by the responder 204. It will be understood that the header is shown in simplified form and may include fields for additional data such as version data, flags, and a message length.

Each of the one or more payloads 232 includes a payload length field and a corresponding payload data field. The payload length field stores the size, e.g. in bytes, of the corresponding payload data. The payload data field stores data that varies depending on a payload type. The payload types included in the message depend upon the mode (e.g. main mode 210, quick mode 212, or user mode 214), state of the negotiation process, and security options employed by the initiator 202 and the responder 204. The different payload types and corresponding payload data are described in **Table**

1, below.

Payload Type	Payload Data
security association (SA)	The security association includes either proposed or agreed upon security parameters.
key exchange data (KE)	Data for a key exchange according to known methods such as a Diffie-Hellman key exchange or elliptical curve.
Main mode nonce (N)	Pseudo random number sent for signing during a main mode exchange.
Quick mode nonce (QmN)	Pseudo random number sent for authentication during a quick mode exchange.
Kerberos authentication data (SSPI)	Kerberos authentication data also referred to as GSSAPI.
Authentication data (AUTH)	A calculated value that incorporates a secret key.
Policy hint (PH)	Data transmitted by a sender that identifies security policy or parameters acceptable to the sender.
Certificate (CERT)	Includes data that establishes a users credentials to another user such as a name, serial number, expiration date, and public key.
Certificate Request (CERTreq)	Request for a network device to provide a certificate.
Identity payload (Id)	Data that identifies a network device, such as an IP address, domain (DNS) name, or fully-qualified domain name (FQDN).
Traffic selector (TS)	Identifies transmitted or received messages subject to a security policy.
Group advertisement (GA)	Identifies a group to which a user belongs.
Vendor Id (V-Id)	Generic data field that includes data to be transmitted from a first network device to a second network device.

Payload Type	Payload Data
Notify	Generic data field that includes data to be transmitted from a first network device to a second network device.

Table 1

The payload types described in Table 1 are identified herein with the subscript "*i*" to represent values associated with the initiator 202 and with the subscript "*r*" to represent values associated with the responder 204 where appropriate. For example, N_i identifies a main mode nonce generated by the initiator 202 and N_r identifies a main mode nonce generated by the responder 204.

Returning to **FIG. 3**, the message 228 may include a plurality of payloads and each payload has different payload data. The payload data is in the form of one of the payload types previously described herein. As shown, a first payload has a payload length 240 and corresponding first payload data 242; a second payload has a payload length 244 and corresponding second payload data 246; and a last payload has a last payload length 248 and corresponding last payload data 250.

The payloads are shown in simplified form and it will be understood that each payload may include additional information, such as data that identifies the payload types included therein.

FIG. 4 illustrates a method for conducting a security negotiation between the initiator 202 and the responder 204 according to the present invention. As previously described, the negotiation is executed by the negotiation module 218 of the initiator 202 and the negotiation module 222 of the responder 204 in accordance with the respective security policies 216 and 220.

The method includes the main mode 210 and the quick mode 212. The main mode 210 and the quick mode 212 are completed through a plurality of messages exchanged between the initiator 202 and the responder 204. Messages 252 and 256 are messages sent from the initiator 202 to the responder 204. Messages 254 and 258 are messages sent from the responder 204 to the initiator 202.

The main mode 210 begins when the initiator 202 sends message 252 to the responder 204. The message 252 has a plurality of payload types including a proposed SA and a main mode nonce (N_i). The message further optionally includes an SSPI payload and key exchange data (KE). As previously described, the proposed SA includes proposed security parameters. The N_i is a pseudo random number generated by the initiator 202. The SSPI is Kerberos authentication data used to authenticate a Diffie-Hellman exchange according to the method described in Piper et al., "A GSS-API Authentication Method for IKE," dated July 14, 2001, which document is hereby expressly incorporated by reference. The use of SSPI is known and, accordingly, is not described in more detail herein.

The responder 204 receives the message 252 and in return sends the message 254 back to the initiator 202. The message 254 has a plurality of payload types including an agreed upon SA, a responder main mode nonce (N_r), a quick mode nonce (QmN_r) and optionally SSPI data and key exchange data (KE). The agreed upon SA includes security parameters, selected from the proposed security parameters, to which the responder 204 agrees. If the responder does not agree to a set of the parameters in the proposed SA, the negotiation fails.

The N_r is a pseudo random number generated by the responder 204. The QmN_r is also a pseudo random number generated by the responder 204. However, the QmN_r is used to facilitate the quick mode 214, while the N_r is used to facilitate the main mode 210. The message 254 is sent as a part of the main mode 210 and also as the beginning of quick mode 212. The SSPI is, as previously described, Kerberos authentication data.

The initiator 202 receives the message 254 and, in return, sends the message 256 to the responder 204. The message 256 has a plurality of payload types including authentication proof (AUTH), SA, traffic selector (TS_i), an initiator quick mode nonce (QmN_i) and optionally key exchange data (KE). The SA in the message 256 includes the security parameters for the quick mode.

The message 256 further optionally includes a certificate request payload (CERTreq) and identification (Id) payloads. The CERTreq payload requests a certificate from the responder. The Id payloads include an Id_i payload that identified the initiator 202. The Id payloads may include an Id_r payload if a previous negotiation attempt was

made between the initiator 202 and the responder 204, but failed because the responder returned a parameter to the initiator that was unacceptable. An example of a parameter that the responder 204 may return which is unacceptable is a certificate. The Id_r payload is sent, from the initiator 202 to the responder 204, in a subsequent negotiation attempt instructing the responder 204 to use a different parameter. For example, the Id_r payload instructs the responder 204 to use a different certificate than was used in the previous negotiation.

The TS_i identifies traffic to be protected according to the initiator's security policy. As an example, the TS identifies traffic by a 5-tuple of source and destination IP addresses, source and destination ports, and protocol type. QmN_i is a pseudo random number generated by the initiator for the quick mode. The AUTH payload is a hash function incorporating a secret key, such as Kerberos secret key, of data previously sent in the messages exchanged between, and known only to, the initiator 202 and the responder 204. The AUTH is used to ensure that there is no attacker between the initiator 202 and the responder 204. The payload types in the message 256 are preferably encrypted using any known suitable method.

After the responder receives the message 256, the main mode 210 is complete. However, the quick mode 212 remains in process. The responder 204 sends message 258. The message 258 has a plurality of payload types including the AUTH, SA, responder traffic selectors (TS_r) and optionally key exchange data (KE) and certificate data (CERT). The AUTH is calculated as previously described. The TS_r is the traffic selectors of the responder, which identifies the traffic to be protected by the responder's security policy. The payload types in the message 258 are preferably encrypted.

The method described with reference to FIG. 3 optionally includes message 260 sent from the initiator 202 to the responder 204 and message 262 sent from the responder 204 to the initiator 202. The messages 260 and 262 each include a notify payload type.

Other processes may exchange messages during the main mode 210 and quick mode 212. For example, an IPSec process executing in the initiator 202 and the responder 204 establish inbound and outbound IPSec SAs. The IPSec SAs define network policy to be used when communicating using a security protocol such as ESP or AH. An inbound IPSec SA at the initiator 202 is established prior to sending message

256, or alternatively, message 260 if the notify payload is sent. An outbound IPSec SA at the initiator 202 is established prior to sending message 258, or alternatively, message 262 if the notify payload is sent. Responder 204 inbound and outbound IPSec SAs are established prior to sending message 258.

After the main mode 210 and the quick mode 212 are complete, the quick mode nonces (N_i, N_r) are used by the initiator and the responder to derive keys according to known techniques. These keys are then used to encrypt traffic using protocols, such as are provided for by IPSec.

As illustrated in **FIG. 4**, the main mode 210 and the quick mode 212 overlap such that messages 254 and 246 include payload types associated with the main mode 210 and the quick mode 212. Because the main mode 210 and the quick mode 212 overlap, the security negotiation is completed with a minimum number of exchanges between the initiator 202 and the responder 204. Additionally, the method provides a mechanism for signaling other processes, such as IPSec, when to perform separate negotiation tasks such as establishing IPSec SAs.

FIG. 5 illustrates an alternate method for beginning the main mode 210. The method provides a way for the initiator 202 to initiate the main mode 210 with the responder 204 when it is unknown whether the responder 204 is capable of conducting a negation according to the invention. If the responder 204 is not capable of conducting a negotiation according to the present invention, the negotiation is carried out using prior methods such as IKE.

The initiator 202 begins the main mode 210 by sending two messages 252 and 264. The message 252 is sent as described with reference to **FIG. 4**. The message 264 is a standard IKE main mode message with a Vendor-Id payload. The Vendor-Id payload includes data indicating that message 252 has also been sent.

The responder receives message 264. If the responder 204 is capable of negotiating according to the present invention, it reads the Vendor Id and from the data learns that main mode message 252 has also been sent. Accordingly, the responder 204 does not respond to message 264. Instead, the responder 204 waits for message 252, assuming the message 252 has not already been received. Once the message 252 is

received, the responder 204 provides response 254 and the negotiation proceeds as described with reference to **FIG. 4**.

If the responder 204 is not capable of negotiating according to the present invention, the responder 204 receives message 252 and does not respond because the responder 204 is unable to interpret the packet. The responder 204 also receives message 264. The responder is able to interpret message 264 except for the Vendor Id, which is ignored. The responder then sends message 266 to the initiator, which message is a standard IKE response. The negotiation proceeds according to the IKE protocol.

FIG. 6 illustrates the user mode 214 according to the present invention. The user mode is optionally conducted following the previously described main mode 210 and quick mode 212. The user mode 214 provides a method of authenticating one or more users. Although the user mode 214 is executed after the quick mode 212, the user mode 214 is completed before the quick mode 212 is activated.

As shown, the user mode 214 includes a first pair of messages 270 and a second pair of messages 272 exchanged between the initiator 202 and the responder 204. The first pair of messages 272 include a first messages 274, sent from the initiator 202 to the responder 204, and a second message 276 sent from the responder 204 to the initiator 202. Each of the messages 274, 276 have a payload type that includes authentication data, which is shown as an SSPI payload by way of example, and not limitation. As previously described, the SSPI payload is Kerberos authentication data. Additional exchanges may occur between the initiator 202 and the responder 204 with additional authentication payloads as needed.

The second pair of messages 272 includes a first message 278, sent from the initiator 202 to the responder 204, and a second message 280, sent from the responder 204 to the initiator 202. Each of the messages 278, 280, have a payload type that includes user authentication data (AUTH), which as previously described is a hash function over previously exchanged data known only to the initiator 202 and the responder 204 that incorporates a secret key.

The method of **FIG. 6** provides a method of conducting user authentication that permits mutual or one way user authentication. As previously described, multiple user modes 214 are possible in conjunction with a single main mode 210.

FIG. 7 and **FIG. 8** illustrate methods according to the present invention that permit dynamic policy discoverability wherein the initiator 202 discovers security parameters that are acceptable to the responder 204. The methods provide a reliable way for the initiator 202 to propose a security association with a set of parameters acceptable to the responder 204.

In the method shown in **FIG. 7**, the initiator 202 sends a message 282 to the responder 204. The message 282 is any suitable data transmitted between devices interconnected by the network 208. For example, the message 282 includes a request to access data of the responder 204. In the example shown, the network policy 216 of the initiator 202 does not require secure communications with the responder 204. Accordingly, the message 282 is sent in clear text, i.e. is not encrypted or otherwise secured.

The network policy 220 of the responder 204 requires secure communication with other network devices, including at least the initiator 202. As a result, the responder 204 is unwilling to respond to the message 282 until secure communications are established between the initiator 202 and responder 204.

The responder 204 sends a response message 284 back to the initiator 202. The response message 284 includes a standard header 230 and payloads 232 (**FIG. 3**). The payloads include a notify payload type and a policy hint (PH) payload type. The notify payload type instructs the initiator 202 that secure communications are required. The notify payload can include any suitable data that identifies the need to conduct secure negotiations.

The PH payload includes data that identifies security parameters acceptable to the responder 204 according to the security policy 220 of the responder. Examples of security parameters identified in the PH payload include acceptable cryptographic mechanisms (e.g. DES or 3DES), keys, lifetime of keys and authentication methods (e.g. Kerberos, Certificates, pre-shared keys).

The response message 284 optionally includes a DOS cookie payload (D-Cookie). The D-Cookie includes the R-Cookie for the responder 204. Providing the initiator 202 with R-Cookie before the security negotiation begins obviates the needs for the responder 204 to maintain state, i.e. store data pertaining to communications with the initiator 202.

Obviating the need to maintain state protects the responder 204 from denial of service attacks as further described in commonly owned co-pending United States Patent Application Serial No. 10/337,763, entitled "Method and Apparatus for Preventing A Denial of Service Attack During Key Negotiation," filed January 7, 2003 which document is hereby expressly incorporated by reference.

The initiator receives the response message 284. In an embodiment of the invention, the initiator 202 includes a stateful firewall filter (SFF) 298. The SFF 298 determines whether inbound messages are allowed to traverse a network stack within the initiator 202 or whether they should be dropped. An example of the SFF 298 is a process that tracks outbound messages from the initiator by source and destination IP address and ports and protocol type. The SFF 298 only allows inbound packets from devices to which communication was initiated by the initiator 202. The SFF 298 helps protect the initiator 202 from malicious network users. For example, the malicious attacker could attempt a denial of service attack on the initiator 202 by sending a large number of packets with false notify payloads using spoofed, i.e fake, IP addresses thereby monopolizing the resources of the initiator 202. An example of the SFF 298 is described in co-pending U.S. Patent Application Serial No. 10/456,770, "A Multi-Layer Based Method for Implementing Network Based Firewalls," filed June 6, 2003, which document is hereby expressly incorporated by reference. In the example, the initiator 202 sent the previous message 282 to the responder. Accordingly, the message 284 is permitted to traverse the network stack within the initiator as a result of being identified by the SFF 298 as a response to message 282.

After the initiator 202 receives the response message 284, it identifies from the notify payload that the responder 204 requires secure communications. The initiator 202 also identifies security parameters that are acceptable to the responder 204 from the PH payload. The initiator 202 then begins the main mode 210 with the responder 204 according to the method described with reference to **FIG. 4**. Preferably, the message 252 sent at the beginning the main mode 210 includes a proposed SA with parameters that match the security parameters in the PH payload.

According to the method of **FIG. 7**, the initiator 202 is able to identify acceptable security parameters according to the responder's security policy 220 by way of the PH

payload thereby increasing the chances of a successful main mode negotiation between the initiator 202 and the responder 204. Additionally, the main mode 210 is initiated by the initiator 202 instead of the responder 204. Accordingly, the roles of the initiator 202 and responder 204 are not subverted.

FIG. 8 also illustrates a method wherein the initiator 202 discovers acceptable security parameters according to the responder's security policy 220. In contrast to **FIG. 7**, however, in the method illustrated in **FIG. 8**, the network policy 216 of the initiator 202 requires secure communications with the responder 204. Accordingly, the security negotiation module 218 of the initiator 202 begins communication with the responder 204 by initiating the main mode 210 as shown by message 286. The message 286, shown in simplified form, has a payload type of the proposed SA.

If the responder 204 agrees to a set of parameters within the proposed SA, the main mode 210, quick mode 212, and optionally user mode 214 are carried out in the manner previously described. If the responder 204 does not agree to a set of parameters within the proposed SA, the responder 204 sends message 288 having the PH payload type. As previously described, the PH payload type identifies security parameters acceptable to the responder 204 in accordance with the responder's security policy 220. This permits the initiator 202 to begin a new main mode 210 by sending a new message to the responder 204 with parameters in the proposed SA that will be acceptable to the responder 204 thereby increasing the chances of a successful negotiation with the responder 204. Alternatively, if the parameters in the PH payload type are unacceptable to the initiator 202, i.e. not according to the security policy 216 of the initiator, the initiator 202 elects not to attempt further communication with the responder 204.

FIG. 9 illustrates a method wherein the initiator 202 and the responder 204 exchange data that defines one or more groups to which the network devices, or users thereof, belong. The initiator 202 sends message 290 with more or more group advertisement (GA) payloads in the first main mode 210 exchange. Each GA payload includes data that identifies a group to which the initiator 202 belongs. The data in the GA payload is not a descriptive name that could be utilized by a malicious user intercepting the packet. Instead, the data in the GA payload is a non-descript, such as an

arbitrary binary number representing the particular group or a hash function of a group name or number along with other data such as nonce and/or time value.

The responder maintains a data structure that identifies authorized groups. When the responder 204 receives the message 290, the responder determines, based on the GA payload, whether the initiator 202 is in an authorized group. If the initiator 202 is not in an authorized group, the responder 204 does not send a reply and remains silent. This prevents the responder 204 from makings its presence known on the network when receiving a message from an unknown network device, which in turn provides added protection against denial of services attacks.

If the initiator 202 is an approved group, the responder 204 sends the message 292 to the initiator 202 with message 292 including the agreed to SA and optionally the responder's own GA payload type identifying a group of the responder. The responder 204 determines which security policy to apply to communications with the initiator 202 based upon the selected group.

FIG. 10 illustrates of method of conducting the main mode 210 and quick mode 212 when the initiator 202 knows the identity of the responder 204 and also knows a public key of the responder 204 before the main mode 210 begins. This may occur, for example, when the responder 204 and the initiator 202 email identity information to each other before the main mode 210 begins.

The method of **FIG. 10** is similar to that shown and described with reference to **FIG. 4**. However, because the initiator 202 has the public key of the responder 204 before the negotiation begins, the initiator 202 can send information in the initial message in an encrypted form thereby preventing access to that information from the malicious user. For example, message 300 is sent from the initiator 202 to the responder 204 with the N_i payload encrypted using the public key of the responder 204. An Id_i payload is also be included within an encrypted payload. The response message 302 from the responder 204 to the initiator 202 includes N_r and N_i payloads encrypted with the initiators public key. The negotiation continues as described with reference to **FIG. 4**.

FIG. 11 illustrates a method 320 used by the initiator 204 to communicate with the responder 204 over the network 208. In step 322, the operating system 134 in the initiator 202 receives a packet or data to be sent from the initiator 202 to the responder

204. The negotiation module 218 determines whether the packet needs to be sent secure, for example according to the ESP or AH protocols, as shown in step 324. The determination of whether to send the packet secure is based upon the security policy 216 of the initiator 202.

If the negotiation module 218 determines that secure communication is not required, the packet is sent unsecured as a standard IP packet as shown in step 326. When the responder 204 sends a return message, the initiator 202 determines whether the responder 204 initiated a security negotiation, as shown in step 328. The responder 204 initiates the security negotiation if the security policy 220 of the responder 204 requires secure negotiation with the initiator 202. The responder 204 initiates communication with message 284. As previously described, the message 284 includes a PH payload that identifies acceptable security policy of the responder 204.

If the responder 204 does not initiate a security negotiation, the process ends as shown and communication between the initiator 202 and responder 204 is conducted using standard IP.

If the negotiation module 218 of the initiator 202 requires secure communication, or if the responder 204 initiates a security negotiation, the negotiation module 218 of the initiator 202 initiates a security negotiation as shown in step 330. The security negotiation is conducted according to the methods previously described herein with references to **FIG. 4- FIG. 9**. The security parameters used by the initiator 202 depend on the security policy 216 of the initiator 202 and any PH payloads previously sent from the responder 204 to the initiator 202.

In step 332, the negotiation module 218 determines whether the security negotiation was successful. If the negotiation was successful, the process ends as shown. If the negotiation was not successful, the negotiation module 218 identifies the reason for the failure as shown in step 334. An unsuccessful security negotiation occurs if the initiator 202 sends proposed SAs to the responder 204 that are not in accordance with the security policy 220 of the responder 204. When that occurs, the responder 204 sends a PH payload in message 288 as previously described. The negotiation also fails when the responder 204 sends an unacceptable parameter, such as a certificate in a CERT payload as previously described.

The initiator 202 then attempts a new negotiation as shown in step 336. The new security negotiation is conducted by beginning a new main mode 210. If a PH payload was received from the responder 204, the initiator 202 sends the first message 252 with a proposed SA payload that conforms to the data in the PH payload. If the negotiation failed because of an unacceptable parameter sent from the responder 204, the initiator sends the Id_r payload in message 256 that notifies the responder 204 to send a different parameter during the new negotiation.

The process 320 continues until a successful security negotiation is completed or alternatively, for a set number of iterations.

All of the references cited herein, including patents, patent applications, and publications, are hereby incorporated in their entireties by reference.

In view of the many possible embodiments to which the principles of this invention may be applied, it should be recognized that the embodiment described herein with respect to the drawing figures is meant to be illustrative only and should not be taken as limiting the scope of invention. For example, those of skill in the art will recognize that the elements of the illustrated embodiment shown in software may be implemented in hardware and vice versa or that the illustrated embodiment can be modified in arrangement and detail without departing from the spirit of the invention. Therefore, the invention as described herein contemplates all such embodiments as may come within the scope of the following claims and equivalents thereof.